SIGNAL DIFFERENTIATION SYSTEM USING IMPROVED NON-LINEAR OPERATOR

CROSS REFERENCE TO RELATED APPLICATIONS

5   This application claims the benefit of priority under 35 U.S.C. § 119(e) to U.S. Provisional application serial no. 60/193,228 filed March 30, 2000.

FIELD OF THE INVENTION

10   The present invention relates to the monitoring of physical systems, processes or machines, and more particularly to systems and methods for discerning signal differentiation using an operator to detect deviations in one or more sets of signals from monitored sensors.

BACKGROUND OF THE INVENTION

15   Generally, there is a need to detect when one or more of a set of signals from sensors monitoring a system, whether a machine, process or living system, deviates from "normal." Normal can be an acceptable functioning state, or it can be the most preferred of a set of various

20   acceptable states. The deviation can be due to a faulty sensor, or to a change in the underlying system parameter measured by the sensor, that is, a process upset.

          While threshold-type sensor alarms have traditionally been used to detect when parameters indicate that a component has strayed away

25   from normal, acceptable or safe operating conditions, many deviations in sensor or underlying parameter values go unnoticed because threshold detection can only detect gross changes. Often such detection may not occur early enough to avoid a catastrophic failure. In particular, there is a critical need to detect when a component, as indicated by a signal or underlying

30   parameter value, is deviating from an expected value, given its relationship to other system components, i.e., in the monitored machine, process or living system. This detection should occur even though the corresponding signal in question is still well within its accepted gross threshold limits.

-1-

Attorney Docket 70478

A number of methods exist that try to use the relationships between sensors, signals, data or the underlying parameters that correspond thereto, to detect notable component changes that otherwise would be missed by "thresholding." Such methods are often data-intensive and computationally demanding. There is a need for accurate empirical modeling techniques that provide computationally efficient and accurate system state monitoring.

SUMMARY OF THE INVENTION

The present invention provides an improved monitoring system and method for ultrasensitive signal differentiation that achieves the detection of changes to and faults in one or more sensor signals in a set that characterizes an underlying "process or machine." The invention achieves accurate results with improved computational efficiency without relying on thresholding. Therefore, less memory and CPU power are needed to perform the necessary calculations. Also, because of the improved computational efficiency, more data "time-slices" can be processed with a given CPU speed. This is useful particularly in systems where signals or data must be sampled at a high rate, or in an application where the CPU or micro-controller is limited.

An empirical model is developed of the process or machine to be monitored, and in real-time sensor data from the monitored process or machine is compared to estimates of same from the model. The results of the comparison are statistically tested with an ultrasensitive difference test that indicates alerts on a sensor-by-sensor basis, thereby providing early warning of process upsets, sensor failures, and drift from optimal operation, long before these would be noticed by conventional monitoring techniques. According to the invention, an improved similarity operator is used in generating the estimates.

The invention provides an improved operator that can be implemented in software on a variety of systems. A typical implementation would be in a programming language like C or C++ on a Unix or Windows workstation, either as a standalone program sampling data via a National Instruments-like pc-card, or as a part or module of a broader process control

software system. The program can be a Windows-like DLL or callable routine, or can comprise an entire suite of presentation screens and settings modification screens. The software can be reduced to a PCMCIA card for addition to a computer with such a port. Then, the data being input to the

5   computation could be fed through a dongle attached to the PCMCIA card. An alternative would be to put the program into microcode on a chip. Input and output of the requisite data to a broader system would depend on the design of the overall system, but it would be well known to circuit designers how to build in the microchip version of this invention. In yet another

10 embodiment, the computation could take place on a server remote (WAN) from the sensors that feed to it. As contemplated in another embodiment, the internet could be used to deliver real-time (or semi-real-time) data to a server farm which would process the data and send back either alarm level data or higher-level messages. In that case, it would become necessary to

15 ensure that the asynchronous messaging "delay" of the internet was of sufficiently unobtrusive to the semi-real-time monitoring taking place over the internet / WAN. For example, bandwidth could be guaranteed so that the asynchronicity of the messaging was not any kind of delay. Alternatively, the sampling rate of the system could be slow enough that the

20 delivery time of Internet messages was negligible in comparison.

Briefly, the present invention relates to a computationally efficient operation for accurate signal differentiation determinations. The system employs an improved similarity operator for signal differentiation. Signals or data representative of several linearly and/or non-linearly related

25 parameters that describe a machine, process or living system are input to the inventive system, which compares the input to acceptable empirically modeled states. If one or more of the input signals or data are different than expected, given the relationships between the parameters, the inventive system will indicate that difference. The system can provide expected

30 parameter values, as well as the differences between expected and input signals; or the system can provide raw measures of similarity between the collection of input signals and the collection of acceptable empirically modeled states.

Attorney Docket 70478

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as the preferred mode of use, further objectives and advantages thereof, is best understood by reference to the following detailed description of the embodiments in conjunction with the accompanying drawings, wherein:

FIG. 1 shows a block diagram of an exemplary laboratory workbench arrangement for gathering process or machine behavior data for distillation;

FIG. 2 shows an example of an embodiment of the present invention in an on-board processor;

FIG. 3 shows an example embodiment wherein a process is shown to be instrumented with sensors having output leads;

FIG. 4 illustrates an implementation of a statistical modeling operator in accordance with the invention;

FIG. 5 shows a method for selecting training set vectors for distilling the collected sensor data to create a representative training data set;

FIG. 6 shows computation of an expected "snapshot," given the real-time actual "snapshot" of the underlying system;

FIG. 7 and FIG. 8 show snapshot numbers 1-6 as deviating signals and noise additive signals respectively with associated vector similarity values using a similarity operator in accordance with the invention;

FIGS. 9A and 9B illustrate a sensor input signal from an air conditioning condenser thermocouple showing positive drift, with FIG. 4A illustrating the residual signal resulting from the drift as deviating over time;

FIGS. 10A and 10B illustrate the residual signal generated in response to negative drift on the sensor input; and

FIGS. 11A and 11B illustrate the introduction of additive noise to the sensor input and the corresponding residual signal.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention is a system, method and program product for monitoring operation of a machine, process, living system or any other such system to accurately predict individual component failures, i.e. component end of life, such that failure or process upset can be anticipated to reduce planned downtime, reduce unplanned downtime and provide for process or product quality optimization.

It is directed to the employment of an improved similarity operator within an empirical modeling engine for use in generating estimates of sensor values for a monitored process or machine based on input of the actual sensor values. Advantageously, the similarity operation carried out in the modeling according to the invention has comparatively low computational needs as compared to other similarity operators that can be used in such an empirical model-based monitoring system, such as is described in US Patent No. 5,987,399 to Wegerich et al., wherein is disclosed a similarity operator employing computationally intensive trigonometric variables. In particular a representative "training" set of data is compared against monitored signal data in a similarity engine and a statistical significance engine. Techniques for achieving ultrasensitive signal differentiation are similar to what is generically described in U.S. Patent No. 5,764,509, directed to the application of a sequential probability ratio test ("SPRT"). Parameter data are gathered from signal sensors monitoring a system such as a machine, process or living system. The number of sensors used is not a limiting factor, generally, other than respecting computational overhead. The present invention is highly scalable. Preferably, the sensors should capture component parameters of at least some of the primary "drivers" of the underlying system. Furthermore, all sensor inputs to the system are best interrelated in some fashion (non-linear or linear).

Turning to FIG. 1, a block diagram of an exemplary laboratory workbench arrangement 100 is shown for gathering process or machine behavior data for distillation. In this example, the monitored system is depicted as a machine prototype 102 and may be, for example, a combustion engine, an electric motor, a pump, a compressor, a refrigerator, and so on. It is understood that, as further indicated herein, the monitored system may be

any machine, living system or system carrying out a process. In this example, the machine 102 is termed a prototype, but importantly, its function is to generate sensor data that is substantially the same as the actual parameter values expected in a production model of the machine, as would be measured by the same sensors. Of course, the training may be in situ wherein the prototype is a production model itself, and ideally, not different in any way from other production models. In addition, when sufficient data has already been accumulated, that previously accumulated data may be used as the training data source, the prototype machine being a virtual machine derived from production machines contributing data to the accumulation.

The machine 102 may be connected to and controlled by a control system 104, generally comprising a microcontroller- or microprocessor-based digital system with appropriate analog/digital and digital/analog inputs and outputs as are known to those skilled in the art. Machine 102 is instrumented with sensors monitoring machine components or reactions thereto (e.g., chamber temperature or pressure) and providing resultant sensor values along outputs 106. During training, the machine 102 is operated through an expected range of operations, and data acquisition system 108 records values of all sensors 106 with which machine 102 is instrumented. Additionally, control signals from control system 104 may also be recorded by data acquisition system 108, and may be used as "sensor signals" that correlate with the other sensor signals.

Data acquired by data acquisition system 108 can, accordingly, be processed using a computer module 110 for producing a distilled training data set representing the operational ranges of machine 102, using the preferred training method, or other appropriate such methods as may be known in the art.

The monitoring system described herein includes an empirical modeling engine and a statistical decision-making engine supported by a suite of software routines for data preconditioning, training, and post-decision reporting. This system is modular and can be applied separately depending on the requirements of the particular monitoring application. Typically, process monitoring equipment employs sensors having some

common characteristics. A sensor data set is acquired as being representative of the normal or desired operation range of the system. The parameters monitored by sensors should be chosen for the model to be correlated, either linearly or nonlinearly. Generally, multiple sensor inputs

5   may be necessary, however, the described algorithms may apply to single sensor applications by using signal decomposition of the single sensor signal into components that can be treated as multiple, correlated inputs for modeling and monitoring. The identification of small deviations in signals from normal operation is provided as indicative of the status of the sensor's

10   associated physical parameter.

The inventive monitoring system can be adapted to provide not just monitoring of existing sensor data, but also generation of derived "virtual" sensor data based on other actually-monitored sensors. Thus, an example 120 of an embodiment of the present invention in an on-board

15   processor is shown in FIG. 2, wherein a system, machine or process, represented by machine 122, is controlled by a control system 124 that is located on the machine. Machine 122 is instrumented with sensors for some of the physical or logical parameters of interest that may be controlling the machine, and the outputs for these sensors are shown as output conductors

20   126, which feed into the control system 124. These are also fed to a processor 128 located within or on the machine, disposed to execute a computing program for monitoring sensor signals and an optional computing program for generating a set 130 of virtual signals on output conductors 126. The processor is connected to a local memory 132, also on

25   or in the machine 122, which stores data comprising the training set distilled to represent the expected operational states of the machine 122. Memory 132 can also advantageously store programs for execution by the processor 128. Virtual signals 130, if included, previously generated by the processor 128 are provided to the control system 124, in lieu of genuine sensor values.

30   Generation of virtual sensor estimates using the improved similarity operator of the present invention can be more fully understood with reference to copending patent application no. 09/718,592 of Wegerich, filed November 22, 2000, and entitled "Inferential Signal Generation for Instrumented Equipment and Process." Virtual signals may be generated as

a cost saving measure to reduce sensor count or for unmonitorable physical or logical parameters of the machine 122.

Processor 128 can also be a part of the control system 124, and in fact can be the processor on which the control system routines are executed, in the event the control system is a digital computed control system. Ideally, the processor 128 and memory 132 are powered by the same power source as the control system 124. However, under certain circumstances, it may also be preferable to provide for a processor 128 and memory 132 independent from another processor and/or memory (not shown) of the control system 124, in order to provide for example virtual signals 130 in a timely fashion, as though they were truly instrumented parameters. For example, processor 128 may operate at a higher clock speed than the control system processor.

FIG. 3 shows an example embodiment 140, wherein a process 142 is shown to be instrumented with sensors having output leads 144. These leads 144 provide sensor signals to a control system 146 that controls the process. These signals are also provided to a remote communications link 148, which is disposed to communicate digital values of the sensor signals to a second remote communications link 150, located at a physically remote place. A processor 152 is provided, which may comprise a computing system and software, that uses the sensor signals received by link 150 to monitor the process 142 for sensor failures, process upsets or deviations from optimal operation and optionally generate at least one virtual sensor signal indicative of an inferred physical parameter of process 142. A memory 154 is provided to store training set data representative of the expected operational behavior of the process 142, according to the distillation method described above.

Furthermore, a display 156 may be provided at the remote location for displaying data descriptive of the process 142, i.e., sensor signals 144 and any virtual signals derived therefrom or both. The virtual signals generated by processor 152 can also be transmitted from link 150 back to link 148 and input over leads 158 to control system 146 for advantageous control of the process. Data from original sensor signals and/or virtual sensor signals (if included) can also be transmitted to a third remote

communications link 160, located at yet a third distant place, for display on display 162, thereby providing valuable information concerning the process to interested parties located at neither the physical process site nor at the monitoring system site.

5   The remote communications links can be selected from a variety of techniques known in the art, including internet protocol based packet communication over the public telecommunications infrastructure, direct point-to-point leased-line communications, wireless or satellite. More specifically, remote links 148, 152 and 160 may be internet-enabled servers

10   with application software for accumulating, queuing and transmitting data as messages, and queues for receiving and reconstituting data arriving as messages. Alternatively, communications can be synchronous (meaning in contrast to asynchronous, message-based communications) over a wireless link.

15   The embodiment of the invention shown in FIG. 3 allows computation of signals using computing resources that are located geographically distant from the system being monitored and/or controlled. One benefit of remote monitoring is that the monitoring and analysis resources may be shared for a multitude of systems, where the memory 154

20   may hold multiple training sets characterizing the various monitored systems, processes and machines or distributed combinations thereof. Another benefit is that results may be displayed and also potentially used in further analysis by interested parties located distant from the system being monitored.

25   A preferred method of employing the improved similarity operator of the present invention is shown in FIG. 4, which illustrates an implementation of the operator according to a method 170 for monitoring a process or machine instrumented with sensors. In a training step 172 a model is derived that captures the normal operating ranges for the sensors.

30   Upon initiating the software system at step 174, training is determined to occur either on-line or not in step 176. Previously archived data 180 may also be used to provide data characteristic of normal or acceptable operation, or data characteristic of normal or acceptable operation may be obtained from the real-time operation of the monitored system 178. According to the

Attorney Docket 70478

method 170, training patterns (the D matrix) are chosen at step 184 to generate the empirical model at 186, as further described below. The training set is a matrix of vectors in which each vector represents one of the normal operating states of the system.

Real time monitoring stage 190 is indicated at step 188, whereupon the model generated at 186 is employed in the steps 192 for estimation. Real-time data is acquired in 192 and an estimate of what the sensor values should be is generated based thereon in view of the model. For each snapshot of data acquired in 192, the similarity operator of the invention is used to compare the actual real-time sensor values to vectors of sensor values in the training set.

The differences between the expected values and real-time sensor signals, i.e., the residual signals formed in 192, are sent to a decision-making engine in the process deviation decision-making step 182. The decision-making engine continuously renders a reliable decision at 194 based on a SSCADI index calculated at 198 over a moving window of successive residual observations, determining whether or not the residual signals reveal a statistically relevant abnormality.

The described embodiment uses a platform with an operating system for real-time monitoring 190 and sampling of the real-time sensor data via a National Instruments feed, with a monitor including an estimation module 192 with a windowing environment to provide feedback in the form of alarms 194 or other high-level messages 196 to an operator of equipment. The program embodying the invention can be written in C or in LabView, a product available from National Instruments Company. The operator portion itself comprises a callable Windows DLL that the LabView software calls. The software may be written as components, including: (1) a training component 172 (for selecting the D matrix from a set previously obtained of "normal" input data), (2) an estimation module component 192 for modeling (to provide the similarity operations both on matrix D and on the real-time input vector, and also to provide an estimated output or an output of the degree of similarity of the vector to the states recognized in the D matrix), and (3) a statistical test component 198 (which tests data such as

using SPRT on the similarity measurement or residual between the input and the estimate).

Monitoring begins by providing signal data as vectors, each with as many elements as there are sensors. A given vector represents a "snapshot" of the underlying system parameters at a moment in time or time correlated. Additional pre-processing can be done, if necessary for time correlation to insert a "delay" between cause and an identified effect between sensors. That is to say, for example, if sensor A detects a parameter change that will be reflected at sensor B three "snapshots" later, the vectors can be reorganized such that a given snapshot contains a reading from sensor A at a first moment, and a corresponding reading from sensor B three snapshots later. Each snapshot is representative of a "state" of the underlying system. Methods for time-correlating in this way are known to those skilled in the art.

Turning to FIG. 5, a method for selecting training set vectors at step 184 is graphically depicted for distilling the collected sensor data to create a representative training data set. In this simple example, five sensor signals 202, 204, 206, 208 and 210 are shown for a process or machine to be monitored. Although the sensor signals 202, 204, 206, 208 and 210 are shown as continuous, typically, these are discretely sampled values taken at each snapshot. As indicated hereinabove, snapshots need not be ordered in any particular order and so, may be ordered in chronological order, parametric ascending or descending order or in any other selected order. Thus, the abscissa axis 212 is the sample number or time stamp of the collected sensor data, where the data is digitally sampled and the sensor data is temporally correlated. The ordinate axis 214 represents the relative magnitude of each sensor reading over the samples or "snapshots."

In this example, each snapshot represents a vector of five elements, one reading for each sensor in that snapshot. Of all the collected sensor data from all snapshots, according to this training method, only those five-element snapshots are included in the representative training set that contain either a global minimum or a global maximum value for any given sensor. Therefore, the global maximum 216 for sensor 202 justifies the inclusion of the five sensor values at the intersections of line 218 with each

-11-

sensor signal 202, 204, 206, 208, 210, including global maximum 216, in the representative training set, as a vector of five elements. Similarly, the global minimum 220 for sensor 202 justifies the inclusion of the five sensor values at the intersections of line 222 with each sensor signal 202, 204, 206, 208, 210.

5   Collections of such snapshots represent states the system has taken on. The pre-collected sensor data is filtered to produce a "training" subset that reflects all states that the system takes on while operating "normally" or "acceptably" or "preferably." This training set forms a matrix, having as many rows as there are sensors of interest, and as many columns

10 (snapshots) as necessary to capture all the acceptable states without redundancy.

Turning to FIG. 6, the training method of FIG. 5 is shown in a flowchart. Data collected in step 230 has N sensors and L observations or snapshots or temporally related sets of sensor data that comprise an array X

15 of N rows and L columns. In step 232, an element number counter i is initialized to zero, and an observation or snapshot counter t is initialized to one. Two arrays "max" and "min" for containing maximum and minimum values respectively across the collected data for each sensor, are initialized to be vectors each of N elements which are set equal to the first column of X.

20 Two additional arrays Tmax and Tmin for holding the observation number of the maximum and minimum value seen in the collected data for each sensor, are initialized to be vectors each of N elements, all zero.

In step 234, if the sensor value of sensor i at snapshot t in X is greater than the maximum yet seen for that sensor in the collected data,

25 max(i) is updated to equal the sensor value and Tmax(i) stores the number t of the observation in step 236. If not, a similar test is done for the minimum for that sensor in steps 238 and 240. The observation counter t is incremented in step 242. In step 244, if all the observations have been reviewed for a given sensor (t=L), then t is reset and i is incremented (to find

30 the maximum and minimum for the next sensor) in step 246. If the last sensor has been finished (i = N), step 248, then redundancies are removed and an array D is created from a subset of vectors from X.

First, in step 250, counters i and j are initialized to one. In step 252, the arrays Tmax and Tmin are concatenated to form a single vector

Attorney Docket 70478

Ttmp having 2N elements. These elements are sorted into ascending (or descending) order in step 254 to form array T. In step 256, holder *tmp* is set to the first value in T (an observation number that contains a sensor minimum or maximum). The first column of D is set equal to the column of X corresponding to the observation number that is the first element of T. In the loop starting with decision step 258, the *ith* element of T is compared to the value of *tmp* that contains the previous element of T. If they are equal (the corresponding observation vector is a minimum or maximum for more than one sensor), it has already been included in D and need not be included again. Counter *i* is incremented in step 260. If they are not equal, D is updated to include the column from X that corresponds to the observation number of T(*i*) in step 262, and *tmp* is updated with the value at T(*i*). The counter *j* is then incremented in step 264. In step 266, if all the elements of T have been checked, then the distillation into training set D has finished, step 266.

Once the D matrix has been determined, in the training phase 172, the preferred similarity engine may begin monitoring the underlying system (122, 142 or 178) and through time, actual snapshots of real sensor values are collected and provided to the similarity engine. The output of the similarity engine can be similarity values, expected values, or the "residual" (being the difference between the actual and expected values).

One or all of these output types are passed to the statistical significance engine, which then determines over a series of one or more such snapshots whether a statistically significant change has occurred as set forth hereinbelow. In other words, it effectively determines if those real values represent a significant change from the "acceptable" states stored in the D matrix. Thus, a vector (Y) of expected values:

$$\vec{y}_{out} = \overline{D} \bullet \vec{W}$$

(i.e., estimates) is determined for the sensors. The expected state vector (Y_{out}) is equal to contributions from each of the snapshots in D, which contributions are determined by the weight vector W. The multiplication operation ($\bullet$) is the standard matrix/vector multiplication operator. W has as many elements as there are snapshots in D. W is determined by:

$$\vec{W} = \frac{\hat{\vec{W}}}{\left(\sum_{j=1}^{N} \hat{W}(j) \right)}$$

$$\overline{W} = \left(\overline{D'} \otimes \overline{D} \right)^{-1} \bullet \left(\overline{D'} \otimes \vec{y}_{input} \right)$$

D is again the training matrix, and D' is the standard transpose of that matrix (i.e., rows become columns). Y_{input} includes the real-time or actual sensor values from the underlying system, and therefore it is a vector

5   snapshot.

The similarity operator $\otimes$ can be selected from a variety of mechanisms for providing a measure of the similarity of one snapshot as compared to another. One general class of similarity operators performs an element for element comparison, comparing a row of the first matrix with a

10   column containing an equal number of elements from the second matrix. The comparison yields a "similarity" scalar value for each pair of corresponding elements in the row/column, and an overall similarity value for the comparison of the row to the column as a whole, e.g., for one snapshot to another.

15   The similarity operator has been improved in the present invention to define the similarity s_i between ith elements as:

$$\theta_i = \frac{\max(x_i, m_i) - \min(x_i, m_i)}{(Max_{range} - Min_{range})}; \quad s_i = 1 - \frac{\theta_i^{\lambda}}{\rho}$$

where:

Max(range) is the maximum value of that "sensor" across the

20   matrix (typically across the transpose of the D matrix), that is the maximum value of x in a given column of x in the above matrix, or in other words the maximum value for that corresponding sensor in the training set;

Min(range) is the minimum value of that "sensor" in the same

25   manner;

X_i = the ith component of the row of the first matrix (x matrix above);

M_i = the ith component of the column of the second matrix (m matrix above);

ρ = an analysis parameter that can be manually varied for enhanced analysis and is defaulted to 1; and

λ = a sensitivity analysis parameter that also can be manually varied for enhanced analysis and is defaulted to 1.

Accordingly, vector identity yields $s_i = 1.0$ and complete dissimilarity yields $s_i = 0.0$. Typically s_i falls somewhere between these two extremes.

Overall similarity of a row to a column is equal to the average (or some other suitable statistical treatment) of the number N (the element count) of s_i values:

$$S = \frac{\sum\limits_{1}^{N} s_i}{N}$$

or alternatively the similarity S of a row to a column can also be calculated:

$$S = 1 - \frac{\left[\left(\sum\limits_{1}^{N} \theta_i\right)\Big/ N\right]^{\lambda}}{\rho}$$

that is, the average of the θ_i, divided by ρ, raised to the power of λ, and subtracted from one. Accordingly, the S values that are summed from θ_i or s_i are the S values depicted in the result matrix above. This operator, having been described generically above, works as follows in the determination of W. The first factor, $D' \otimes D$, is referred to herein as matrix G, i.e.,

$$\overline{G} = \overline{D}' \otimes \overline{D}$$

So, for example, with D and its transpose for 4 sensors a, b, c, d and for n training set snapshots are:

$$\overline{D}' = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ a_n & b_n & c_n & d_n \end{bmatrix} \qquad \overline{D} = \begin{bmatrix} a_1 & a_2 & . & . & . & . & . & a_n \\ b_1 & b_2 & . & . & . & . & b_n \\ c_1 & c_2 & . & . & . & . & c_n \\ d_1 & d_2 & . & . & . & . & d_n \end{bmatrix}$$

then the matrix G is:
$$\overline{G} = \begin{bmatrix} g_{11} & g_{12} & . & . & g_{1n} \\ g_{21} & g_{22} & . & . & g_{2n} \\ . & . & . & . & . \\ . & . & . & . & . \\ g_{n1} & g_{n2} & . & . & g_{nn} \end{bmatrix}$$

5   where, for example, element g_{12} in the matrix G is computed from row 1 of D-transpose (a_1, b_1, c_1, d_1) and column 2 of D (a_2, b_2, c_2, d_2) as either:

$$S_{12} = \frac{\sum_{i=a,b,c,d} s_i}{4} \qquad or \qquad S_{12} = 1 - \frac{\left[\left(\sum_{i=a,b,c,d} \theta_i \right) \big/ 4 \right]^{\lambda}}{\rho}$$

where the elements s_i and θ_i are calculated as:

$$\theta_a = \frac{\max(a_1,a_2) - \min(a_1,a_2)}{(Max_a - Min_a)};$$
$$\theta_b = \frac{\max(b_1,b_2) - \min(b_1,b_2)}{(Max_b - Min_b)};$$
$$\theta_c = \frac{\max(c_1,c_2) - \min(c_1,c_2)}{(Max_c - Min_c)};$$
$$\theta_d = \frac{\max(d_1,d_2) - \min(d_1,d_2)}{(Max_d - Min_d)};$$

$$and \qquad s_i = 1 - \frac{\theta_i^{\lambda}}{\rho}$$

-16-

The resulting matrix G is symmetric around the diagonal, with ones on the diagonal. To further calculate W, the computation of $(D' \otimes Y_{input})$ is performed in a similar fashion, except Y_{input} is a vector, not a matrix:

$$\overline{D}' \otimes \overline{y}_{input} = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ a_n & b_n & c_n & d_n \end{bmatrix} \otimes \begin{bmatrix} a_{in} \\ b_{in} \\ c_{in} \\ d_{in} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ . \\ . \\ . \\ S_n \end{bmatrix}$$

5   The G matrix is inverted using standard matrix inversion, and the result is multiplied by the result of $D' \otimes Y_{input}$. The result is W:

$$\overline{W} = \overline{G}^{-1} \bullet \begin{bmatrix} S_1 \\ S_2 \\ . \\ . \\ S_n \end{bmatrix} = \begin{bmatrix} W_1 \\ W_2 \\ . \\ . \\ W_n \end{bmatrix}$$

These weight elements can themselves be employed, or expected values for the sensors can be computed, applying the training matrix D to the weight

10   vector W:

$$\overline{y}_{expected} = \overline{D} \bullet \overline{W} = \begin{bmatrix} a_1 & a_2 & . & . & a_n \\ b_1 & b_2 & . & . & b_n \\ c_1 & c_2 & . & . & c_n \\ d_1 & d_2 & . & . & d_n \end{bmatrix} \bullet \begin{bmatrix} W_1 \\ W_2 \\ . \\ W_n \end{bmatrix} = \begin{bmatrix} a_{ex} \\ b_{ex} \\ c_{ex} \\ d_{ex} \end{bmatrix}$$

Turning to the analysis or tuning parameters λ and ρ, while either value can remain set to its default value of 1, analysis parameter ρ can be varied to all values greater than or equal to 1. The sensitivity analysis parameter λ can be

15   varied to any value and, preferably, to any positive value and more particularly, can be selected in the range of 1-4.

The computation results of the expected sensor values is subtracted from the actual sensor values to provide a "residual". The residual is also a vector having the same number of elements as the sensor-count, with each element indicating how different the actual value is from the estimated value for that sensor.

So, for example, set forth below is an example of C code for the overall functionality of the inventive similarity operator based upon a modeling technique for the calculation of the similarity between two vectors.

First user specified selectable parameters include:

D: Training matrix; M rows by N columns, where M is the number of variables in the model and N is the number of training vectors chosen from a training data set.

λ: Non-linearity exponent, i.e., an analysis or coarse tuning parameter.

ρ: Scaling factor, i.e., an analysis or fine tuning parameter.

Then, inputs include:

D: The two dimensional training matrix.

M: The number of variables or rows in the D matrix.

N: The number of training vectors or columns in the D matrix.

lambda: OP Non-linearity (coarse) tuning parameter.

pRow: Scaling (fine) tuning parameter.

Outputs are defined as:

R: A range vector defined externally to avoid dynamic memory allocation.

G: The model matrix G calculated as set forth below.

Before real-time monitoring can begin, the global range R for each of the $i = 1,2,...,M$ variables in the model must be calculated using a representative data set as described hereinabove with reference to Figures 5 and 6. If X is a data matrix with rows corresponding to observations and columns to variables, then the range is calculated for each variable as follows.

-18-

$$"R_i = \max(X(n,i)) - \min(X(n,i)), \text{ over all } n"$$

```
for (column=0; column<M; ++column)
{
        min = X[column];
        max = X[column];
        for (row=1; row<N; ++row)
        {
                if (X[row*M + column] < min)  min = X[row*M + column];
                if (X[row*M + column] > max)  max = X[row*M + column];
        }
        R[column] = max - min;
}
```

Then after determining the ranges R_i for each variable, the OP operator model matrix G can be constructed from an appropriate training matrix D. The model matrix is an N by N square matrix calculated using the OP operator, as follows:

$$"G = D^t \otimes_{SSCOP} D, (where, \otimes_{SSCOP} \text{ is the SSCOP operator})"$$

The SSCOP operator measures the similarity between vectors. When calculating G, the similarities between all pairs of column vectors in D are calculated. The SSCOP operator uses the ranges R_i, non-linearity analysis parameter λ, and the scaling analysis parameter ρ to calculate similarity measurements. Performing the following procedure carries out the similarity between two vectors, d_1 and d_2 in D using SSCOP.

If D contains N column vectors, each including M elements,

$$D = \left[d_1 \mid d_2 \mid \cdots \mid d_N \right]$$
$$d_k = \left[d_1(t) \, d_2(t) \quad \cdots \quad d_M(t) \right]$$

then vector similarity is measured as follows:

i) Calculate the elemental similarity angles θ_i for the ith sensor and each pair of elements in d_1 and d_2.

$$\theta_i = \frac{max(d_1(i), d_2(i)) - min(d_1(i), d_2(i))}{R_i}$$

ii) Calculate the elemental similarity measurements.

5

$$s_i = 1 - \frac{\theta_i^{\lambda}}{\rho}$$

iii) Next, calculate the overall vector similarity.

$$first, if\ (s_i < 0),\ then\ s_i = 0$$

$$S = \frac{1}{M} \sum_{i=1}^{M} S_i$$

These steps are used to calculate the similarity between all combinations of vectors in D to produce G.

10   So, for example:

Given a data set X, the Range of each variable is

X = [[2.5674   -1.1465   0.3273

1.3344   1.1909   0.1746

3.1253   1.1892   -0.1867

15   3.2877   -0.0376   0.7258]] ; .

Setting $\rho = 2$, $\lambda = 3$, and D is

D = [[2.5674   1.3344   3.1253   3.2877

-1.1465   1.1909   -1.1892   -0.0376

0.3273   0.1746   -0.1867   · 0.7258]] ;

20   the resulting model matrix G is then

G = [[1.0000   0.7906   0.8000   0.9600

0.7906   1.0000   0.8612   0.7724

0.8000   0.8612   1.0000   0.8091

0.9600   0.7724   0.8091   1.0000]] ;

25   and the inverse is (condition number equals 91.7104):

-20-

$$Gi = [[\quad 13.8352 \quad\quad -1.8987 \quad\quad 0.3669 \quad\quad -12.1121$$
$$-1.8987 \quad\quad 4.3965 \quad\quad -2.8796 \quad\quad 0.7568$$
$$0.3669 \quad\quad -2.8796 \quad\quad 4.8407 \quad\quad -2.0446$$
$$-12.1121 \quad\quad 0.7568 \quad\quad -2.0446 \quad\quad 13.6974 \ ]] ; .$$

```
//  Next, calculate G from Dt SSCOP D
for (row=0;  row<N;  ++row)
{
    for (column=0;  column<N;  ++column)
    {
        G[row*N + column] = 0;
        for (i=0;  i<M;  ++i)
        {
            if (D[i*N + row] > D[i*N + column])
            {
                theta = (D[i*N + row] - D[i*N + column]) / R[i];
            }
            else
            {
                theta = (D[i*N + column] - D[i*N + row]) / R[i];
            }
            s = 1 - pow(theta, lambda) / pRow;
            if (s < 0) s = 0;
            G[row*N + column] += s;
        }
        G[row*N + column] /= M;
    }
}
```

A statistical significance test is applied to the elements of the residual, or to the residual as a whole. More particularly, a test known as the sequential probability ratio test ("SPRT") is applied to the residual for each sensor. This provides a means for producing an alarm or other indicator of statistically significant change for every sensor in real-time.

The SPRT type of test is based on the maximum likelihood ratio. The test sequentially samples a process at discrete times until it is capable of deciding between two alternatives: $H_0:\mu=0$; and $H_A:\mu=M$. It has been demonstrated that the following approach provides an optimal decision method (the average sample size is less than a comparable fixed sample test). A test statistic, Ψ_t, is computed from the following formula:

$$\psi_t = \sum_{i=1+j}^{t} \ln\left[\frac{f_{H_A}(y_i)}{f_{H_0}(y_i)}\right]$$

where $\ln(\cdot)$ is the natural logarithm, $f_{H_s}()$ is the probability density function of the observed value of the random variable Y_i under the hypothesis H_s and j is the time point of the last decision.

In deciding between two alternative hypotheses, without knowing the true state of the signal under surveillance, it is possible to make an error (incorrect hypothesis decision). Two types of errors are possible. Rejecting H_0 when it is true (type I error) or accepting H_0 when it is false (type II error). Preferably these errors are controlled at some arbitrary minimum value, if possible. So, the probability of a false alarm or making a type I error is termed α, and the probability of missing an alarm or making a type II error is termed β. The well-known Wald's Approximation defines a lower bound, L, below which one accepts H^0 and an upper bound, U beyond which one rejects H_0.

$$U = \ln\left[\frac{1-\beta}{\alpha}\right]$$

$$L = \ln\left[\frac{\beta}{1-\alpha}\right]$$

Decision Rule: if $\Psi_t < L$, then ACCEPT H_0;

else if Ψ, $< U$, then REJECT H_0; otherwise, continue sampling.

To implement this procedure, this distribution of the process
must be known. This is not a problem in general, because some a priori
information about the system exists. For this purpose, the multivariate
Normal distribution is satisfactory.

For a typical sequential test

$$\psi_{t+1} = \psi_t + \overline{M1} \sum {}^{-1} \left(\overline{y}_{t+1} + \frac{\overline{M1}}{2} \right)$$

where M is the system disturbance magnitude and $\overline{Y}$ is a system data vector.

FIG. 7 shows deviating signals for snapshots numbered 1-6
with associated vector similarity values using the inventive similarity
operator. These six separate examples show vector-to-vector similarity
graphically depicted. In each snapshot chart, the x-axis is the sensor
number, and the y-axis is the sensor value. Each snapshot comprises 7
sensor values, that is, seven elements. The two lines in each chart show the
two respective snapshots or vectors of sensor readings. One snapshot may
be from real-time data, e.g., the current snapshot, and the other snapshot
may be from the D matrix the comprises the model. Using the inventive
similarity operator described above, element-to-element similarities are
computed and averaged to provide the vector-to-vector similarities shown
in the bar chart in FIG. 7. It can be seen that the similarity comparison of
snapshot chart #6 renders the highest similarity scalar, as computed using
the similarity operator of the present invention. These similarity scalars are
used in computing estimates as described above, with regard to the
generation of W.

FIG. 8 shows noise additive signals for snapshots numbered 1-
6 with associated vector similarity values using the inventive similarity
operator. The output of the similarity engine can be similarity values,
expected values, or a "residual," i.e., the difference between the actual and
expected values. A more typical set of examples of sensor data with noise

are depicted in six snapshot charts, this time comprising 40 sensor elements. Each chart again contains two lines, one for each of two snapshots being compared using the similarity operator of the present invention, these charts are far more difficult to compare visually with the eye. The similarity

5   operator scalar results are shown in the bar chart of FIG. 8, which provides an objective measure of which vector-to-vector comparison of the set of six such comparisons actually has the highest similarity. Again, the similarity values can be output as results (e.g., for classification purposes when comparing a current snapshot to a library of classified snapshots) or as input

10   to the generation of estimates according to the equations described above, which ultimately can be used for monitoring the health of the monitored process or machine.

FIGS. 9A and 9B illustrate a sensor input signal from an air conditioning condenser thermocouple showing positive drift, with FIG. 9A

15   illustrating the residual signal resulting from the drift as deviating over time. In FIG. 9A, it can be seen that the drifted "actual" sensor signal and the estimated signal generated using the similarity operator of the present invention generate a difference or "residual" that grows in FIG. 9B as the estimate and actual diverge. The estimate clearly follows the original pre-

20   drift sensor value that the drifted sensor was based on.

FIGS. 10A and 10B illustrate an example of a residual signal generated in response to negative drift on the sensor input. Because the drift is in the opposite direction, it is negative and the estimate again follows what the sensor should have been. Note that the residual shown in FIG. 10B

25   is actually growing in the negative direction.

FIGS. 11A and 11B illustrate an example of the introduction of additive noise to the sensor input and the corresponding residual signal. Note that the failing sensor is actually getting noisier, and the similarity estimate remains close to the pre-failed sensor signal on which the noisy

30   signal was based. The residual in FIG. 11B swings wider and wider, which would be easily detected in the SPRT-based statistical decision engine and alarmed on.

Thus, the present invention provides a superior system modeling and monitoring tool for ultra sensitive signal differentiation accurately detecting system parametric deviations indicative of more subtle underlying system or system component changes, whether the system is a machine, a process being carried out in a closed system, a biological system or any other such system. The tool differentiates between anomalous results caused by defective sensors and component related failures, allowing adjustment therefor. Further, the signal differentiation sensitivity of the tool far exceeds prior art thresholding results using less memory and computer power to consider larger volumes of data, interactively if desired, irrespective of available computer power.

It will be appreciated by those skilled in the art that modifications to the foregoing preferred embodiments may be made in various aspects. The present invention is set forth with particularity in the appended claims. It is deemed that the spirit and scope of that invention encompasses such modifications and alterations to the preferred embodiment as would be apparent to one of ordinary skill in the art and familiar with the teachings of the present application.